DALI USER'S MANUAL

P. van der Wolf
J. Liedorp

Delft University of Technology
Dimes Design and Test Centre
Feldmannweg 17, P.O. Box 5053, 2600 GB Delft
The Netherlands

Abstract

This manual describes the use of *dali*, the interactive layout editor of the Nelsis IC Design System. It starts with an overview of the general features of dali: graphics capabilities, support for hierarchical design, etc. Thereafter, the individual commands are discussed in more detail.

Copyright © 1987 by the author(s).
All rights reserved.

Date: November, 1987.
Revision: September, 1988.
Revision: December, 1990, by ESE Electronic Systems Engineering B.V.
Revision: January, 1993, by S. de Graaf.

1. INTRODUCTION

1.1 Running Dali

Dali (the Delft Advanced Layout Interface) is an interactive front-end to the layout database of the Nelsis IC Design System, to be used for the creation and manipulation of layouts of integrated circuits. It has been written in the programming language C and runs under the DOS and Unix operating system.

There are two versions of the program which have virtually the same functionality to the end-user but differ in their graphics capabilities. One version is invoked as 'dali'. It uses the Graphical Kernel System (GKS) for device independent graphics. As a consequence, it runs on a variety of graphics devices. The other version is invoked as 'Xdali', to be run in a X Window System graphics environment.

Dali must be started in a *project* directory. See also *mkpr* (1ICD). It may be invoked in the following ways:

dali [-**f**] [-**o**] [-**r**] [-**d** device] [-**g** graphics] [cell_name]
Xdali [-**A**] [-**C**] [-**f**] [-**o**] [-**r**] [-**h** host:d#[.s#]] [=geo] [cell_name]

The meaning of the options is:

-f With this option one may specify an alternate dali design-rule file "dali_drc", which must reside in the current working directory (project directory).

-o When this option is selected, dali writes messages to an "animate" file in some occasions when reading the ".dalirc" setup file.

-r This option may be selected to run dali in projects where only 'read' access is permitted. Cells can then be read for display purposes, but dali will not allow edited cells to be written to the project database.

-d device GKS-version only. With the name 'device' one may choose the pathname of the graphics device one wants to use. For example, used in combination with the 'hpgp' graphics device.

-**g** graphics GKS-version only. With the name 'graphics' one may choose the kind of graphics terminal one wants to use. It depends on the system one is working on, which graphics terminals are supported.
Examples are: apollo_15_color, hpgp, rast10, hp2397, hp98700.

If neither the **-d** nor the **-g** option is specified for the GKS-version, the environment variable TERM is used to obtain the kind of graphics terminal one is working on. If this is not a terminal supported by dali an error message is generated.

-A Xdali only. With this option a plane mask for the Text Graphics Context can be set, used for Apollo DN3000 workstations.

-C Xdali only. With this option the use of the standard NELSIS ICD colormap can be switched off. See also *setcmap* (1ICD).

-h host:d#[.s#] Xdali only. With this option one may specify which display is to be used by Xdali. The display is identified by a hostname 'host' and a display number 'd#', separated by a colon. If this does not completely identify a unique screen, an additional screen number '.s#' has to be specified. For example: "`dutentn:0.0`".

If this option is not used for Xdali, the environment variable DISPLAY is consulted to identify the display.

=geo Xdali only. The window geometry 'geo' specifies the initial window size and placement according to the standard X Window geometry format <width>x<height>{+-}<xoffset>{+-}<yoffset>. For example, =+0 gives a default window at xoffset 0, i.e. in the upper left corner of the screen. The window geometry =1000x800+0 gives a rather big window in the upper left corner, and =1000x800-0+0 gives a window of the same size in the upper right corner.

cell_name Name of cell that is to be read at start-up.

Since dali and Xdali differ primarily in the graphics interface employed but have virtually the same functionality, we will simply refer to both versions as 'dali' when describing this functionality in the following sections.

For the Xdali version some X Window System parameters can be set via the ".Xdefaults" setup file. This are the *BorderWidth* (default: 1), the *FontName* (default: "fixed"), and the window *GeoMetry* (default: --). Note that the window geometry can also be specified via the command-line. Example of ".Xdefaults" entries:

```
Xdali.BorderWidth: 2
Xdali.GeoMetry: 850x700+20+20
```

1.2 The Screen Layout

Dali divides the screen area of the graphics terminal into four parts, called viewports:

1. MENU viewport: the left side of the screen area.
   In this viewport the *commands* available are displayed. One may select a command by pointing at it (section 1.3). The corresponding area is highlighted until the operation is finished or another command is selected. Certain commands require additional information to be specified during their execution. On these

occasions dali can use this viewport to present alternatives from which the user has to choose. Selection proceeds in the same way as command selection.

2. LAYER viewport: the strip at the bottom of the screen area.
   In this viewport the *layers* one can use in the *process* of the current project are given (section 1.8). One may select or deselect a layer by pointing at the rectangle with the name of the layer in it. If a layer is selected (activated) the corresponding rectangle is painted in the color of the layer. An exception is made for the layers that have the background color (black), typically contact hole layers; their (non-filled) rectangle is painted white upon selection. The active layers may be used in subsequent operations.

   Via the ".dalirc" setup file (*disabled_layers* command) layers may be declared to be non-editable. Non-editable layers are indicated by a white cross in the layer rectangle, and can not be selected.

   Via the visible menu or the ".dalirc" setup file (*visible* command) layers may be declared to be non-visible. Non-visible layers are indicated by a white 'V' in the layer rectangle, and can not be selected.

3. TEXT viewport: the strip at the top of the screen area.
   In this viewport dali displays (error) messages, or asks the user to enter some alphanumerical information such as a cell name. In the latter case dali also uses this viewport to echo the characters that are entered via the keyboard.

   In the Xdali version the rightmost part of TEXT viewport is used also by the tracker to display cursor coordinates and displacements.

4. PICT viewport: the large center area.
   In this viewport the picture of the layout one is working on is displayed and points can be entered if required by one of the commands.

The different viewports can be recognized in Figure 1.1, which is the real-life appearance of dali.

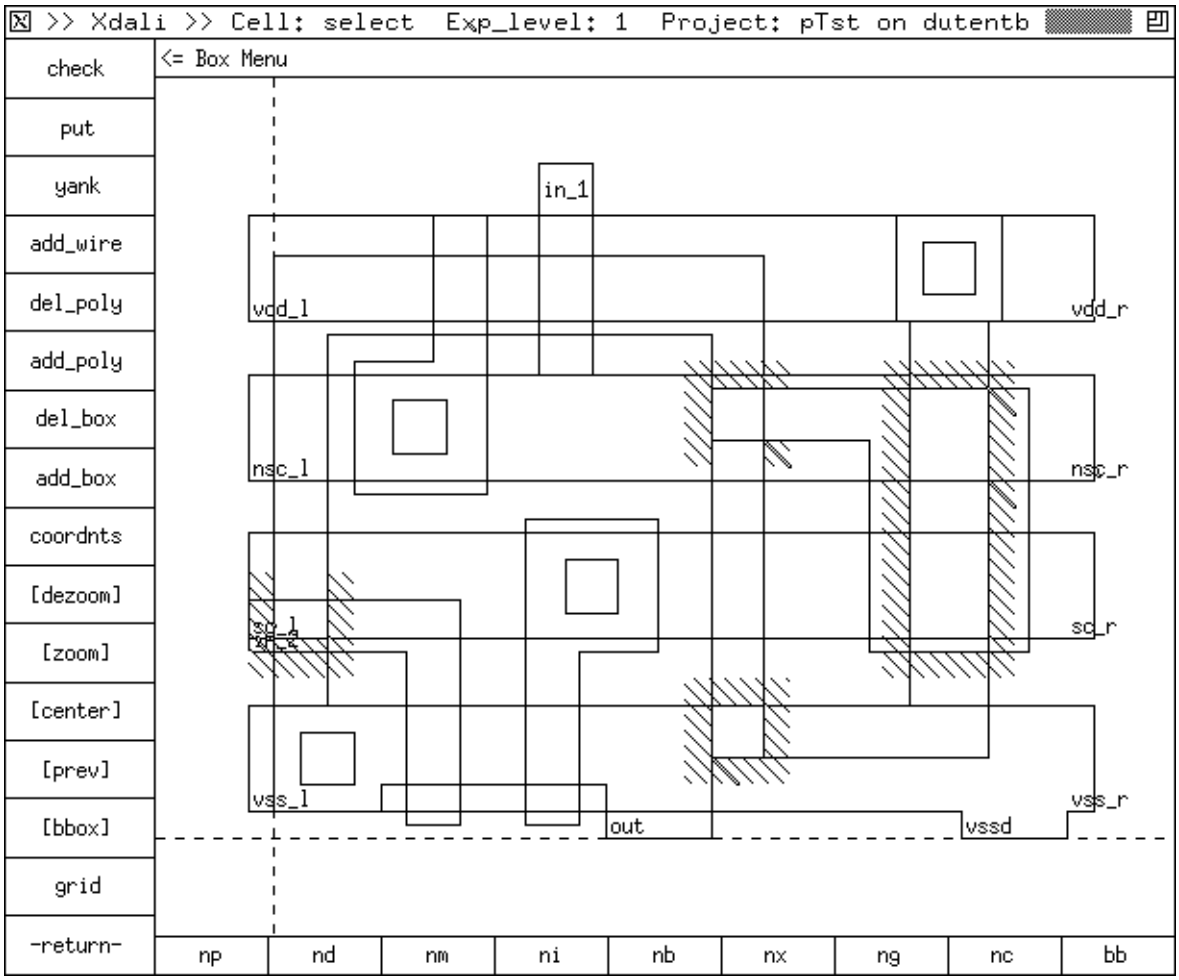

Figure 1.1. Appearance of dali with a layout being displayed

1.3 Pointing and Cursors

Dali accepts input from the keyboard of the terminal and from a mouse or tablet attached to it. Whenever a point has to be entered, dali displays a *cursor* on the screen, which can be moved around with the aid of the *mouse* (or stylus of the tablet). Upon pushing the button of the mouse (or pressing the stylus) the point is selected. In GKS this is typically the rightmost button on a 3-button mouse. In the sequel of this manual we will refer to this activity as *pointing*.

Dali uses several types of cursors. For most operations, e.g. command and layer selection, a small *crosshair* is used. When rectangular areas, polygons or center-lines of wires have to be entered, dali switches to *rubber-box* / *rubber-line* cursors after the first point has been fixed.

On occasions where dali expects the user to specify a point in the PICT viewport, it is often allowed to point at a command or layer instead. This gives the user the opportunity to cancel the command in progress by selecting the same command again or another command, or to (de)select the layers on which the command will operate. Examples are the *add_box* and *add_wire* commands.

The Xdali version provides a tracker which displays coordinate information of the cursor in the rightmost part of the TEXT viewport as the cursor is moved around in the PICT viewport. For rubber-box / rubber-line cursors the tracker displays the current cursor position as well as the displacement from the start(last end) point. Via the settings menu the tracker mode can be set "auto", "on" or "off" with the *tracker* command. The default tracker mode ("off") can also be changed via the ".dalirc" setup file. In "auto" mode the tracker information is only visible if a start point is specified and the cursor is switched to rubber-box / rubber-line cursor.

1.4 Cell Editing

Dali must be started in a *project* directory. It then interacts with the layout database of this project to allow the designer to create or modify layout *cells*. These cells are the entities on which dali operates; they are retrieved / stored as a whole from / into the database.

While operating on a cell, all modifications are performed on a copy which dali maintains internally in its *private workspace*. Existing cells can be *read* from the database into this workspace with the *read_cell* command. Changes that have been made on the workspace copy are effectuated only when the cell is *written* explicitly to the database with the *write_cell* command. Thus, one should take care to *save* the workspace copy whenever a considerable amount of valuable changes has been made to it. The workspace can be *erased* with the *erase_all* command. This will *not* affect an original or previously saved cell description that is already present in the database.

Cells may be created from scratch or by modifying cells that were already present in the database. In the former case the newly created cell can be stored in the database under a new name. In the latter case the cell can either be stored under its present name, thereby overwriting the old cell description, or it can be stored under a new name. There is a setup option *use_new_name*, whenever this option is activated the new name is used in place of the old name for the edited cell in the workspace after the *write_cell* command. In the sequel of this manual we will use the term "edited cell" to refer to the piece of design that is present in the private workspace of dali, irrespective of its origin.

A layout cell consists of three types of data:

- *primitive mask geometries* (representation data).
- *terminals* (interface data).
- *instances*: references to component cells (hierarchical data).

Dali provides support for the manipulation of all three types of data. For each type a set of commands is available in a separate menu. In the following three sections we describe dali's capabilities in handling these types of data.

1.5 (Non-) Orthogonal Mask Geometries

The primitive mask geometries of the edited cell can be manipulated by adding / deleting *boxes* or *polygons* or by adding *wires* for the selected layer(s). As opposed to several other layout editors, manipulation of the primitive mask geometries with dali is *not* element-based. Instead, a *cut-and-paste* technique is supported. This implies, for instance, that a rectangular mask geometry that was once added as a box does not have to be deleted as a whole; parts of it can simply be cut away.

By means of *buffer* operations, chunks of mask geometries can be moved or copied. Deleted geometries are saved in the buffer, allowing the designer to undo the delete operation by *putting* (adding) the buffer at the same spot, or *move* the geometries by putting the buffer somewhere else. The buffer can also be filled explicitly, without deleting geometries, thereby providing a *copy* facility. Care should however be taken, as only *one* geometry buffer is available; whenever it is filled, the old contents are destroyed.

Dali supports orthogonal (Manhattan) and 45 degree geometries only. The box (or rectangle) is the most frequently used type of geometry. Boxes can be added / deleted with the help of a rubber-box cursor. The corner points are automatically rounded to the *lambda grid* or a multi-lambda *snap grid*, if active (section 1.10).

Polygons having edges at 45 degree angles can be added / deleted with the help of a rubber-line cursor to specify the contour of the polygon. Dali can handle self-intersecting polygons. Corner points may be entered on grid points and half grid points. (As intersection points of 45 degree edges may now occur at quarter grid points, dali internally maintains a higher precision than the *lambda* grid presented to the user.) This capability is offered to allow the user to create wires containing 45 degree parts with a width that only slightly deviates from the width of the orthogonal parts. The snap grid does not apply to polygon editing.

A wire facility is included to permit convenient specification of interconnection patterns. After a width has been fixed, the center-line of the wire can be entered with the help of a rubber-line cursor. Both orthogonal wires and wires containing 45 degree parts can be

entered this way. The center-line is automatically fixed on the *lambda grid* or a multi-lambda *snap grid*, if active (section 1.10). If the snap grid is active only even widths are permitted.

As dali is capable of handling Manhattan and 45 degree geometries only, one should take care when reading cell descriptions from the database that have been generated by other programs and contain arbitrary geometries (polygons with non-45 degree edges, circles). Dali checks all layout elements when reading from the database. All edges are checked to be proper orthogonal or 45 degree edges. Whenever a check fails the element is skipped and a warning is generated.

1.6 Terminals

A terminal is a named box, which may be arrayed. Terminals play an important role in the layout verification strategy of the Nelsis IC Design System. The terminals of a cell specify at which points the cell may be connected to its surroundings upon instantiation (section 1.7).

Dali offers commands to add or delete terminals. When the terminals in a specified area are deleted (*del_area*), they are saved in a terminal *buffer* (to be distinguished from the geometry buffer), which can be placed again at a later time (*put_buf* command). Note that the terminal buffer is overwritten after each successful *del_area* or *yank_area* operation.

Other commands allow terminals to be *moved* or *arrayed*. These commands operate on the terminal, which is selected with the cursor. As an example of a higher level assembly function, dali has the ability to generate terminals "on top of" the terminals of the cell's instances, thereby making these spots available for interconnection at the next higher hierarchical level. This is possible with the lift terminals (*lift_tms*) command.

On various occasions one has to identify the terminal on which a command should operate (*array_term*, *del_term*, etc.). A terminal can be selected by just pointing at it in the PICT viewport. In case the terminal has been arrayed, one can identify it by pointing at one of its occurrences. If the point is ambiguous because it identifies more than one terminal, dali asks for a further specification by presenting a sub-menu with the names of the terminals that were identified by the specified point. In this case, one of the names has to be selected from the sub-menu. The name of the selected terminal is reported in the TEXT viewport.

1.7 The Cell Hierarchy

1.7.1 Instances and Instance Manipulation

Dali supports hierarchical design, meaning that cells can be constructed using other (smaller) cells as their components. The definition of these component cells must be

present in the database before they can be *instantiated* in the edited cell. An *instance* is a *reference* to a component cell, accompanied by a geometrical transformation (translation and orientation) and repetition information. Dali offers commands to add or delete instances to / from the edited cell, thereby changing its hierarchical composition. Normally this are *local* cells, but these may also be instances of cells that have previously been *imported* from other projects. See also *impcell* (1ICD).

The translation and orientation of the instances in the edited cell can be manipulated by *move, rotate* and *mirror* commands. *Arrays* of instances can be created / manipulated by commands that allow the user to specify repetition numbers and distances. These instance manipulation commands operate on the instance, which is selected with the cursor.

Dali assumes the component cells to be fixed; it does not allow the definition of a component cell to be changed from *within* the edited cell. Only the transformation and repetition information of the reference can be manipulated. Whenever changes have to be made to the component cell itself, its definition must be edited separately.

By default, dali displays only the bounding box of an instance, with the name of the instance and the name of the component cell displayed in the lower left corner. Note that the name of the cell is displayed between '(...)' and that the instance name is displayed above the cell name. This instance name is a dot '.', if no instance name is given to this cell occurrence. Use the name instance (*name_inst*) command to give this instances a instance name. Set *ask_inst* (in the settings menu) to do it directly by the *add_inst* command. In case of an array of instances, dali shows the bounding boxes of all the occurrences, with the name of the instance and the name of the component cell in the (0, 0) occurrence, together with the [n × m] array dimensions. When a command for the manipulation of the translation or orientation of an instance is executed on an array of instances, this will affect all individual occurrences; Dali handles the array as one entity.

1.7.2 Selection of Instances

On various occasions one has to identify the instance on which a command should operate (*name_inst, del_inst, indiv_exp, lift_tms*). An instance can be selected by just pointing at a position within its bounding box (repetitions included) in the PICT viewport. If the point is ambiguous because it lies inside the bounding box of more than one instance, dali asks for a further specification by presenting a sub-menu with the names of the component cells of these instances. In this case, one of the names has to be selected from the sub-menu. The name of the component cell of the selected instance (and its instance name) is reported in the TEXT viewport. If this is the correct one, then this cell must be selected again, else another cell must be selected.

1.7.3 Expansion of Instances

Dali offers commands that enable the user to make the contents of the instance(s) visible up to a certain *expansion level*, either by setting a *global* expansion level for *all* instances or by setting the level on an *individual* basis (*indiv_exp* command). The default

situation, where only the bounding box of an instance is displayed, is called level 1. If an instance is *expanded* to level 2, the primitive mask geometries of the component cell are displayed, together with bounding boxes for the instances that are in turn part of this component cell. As the expansion level is further increased, say to level n, dali displays the mask geometries of cells from deeper levels in the hierarchy, up to level n. Dali always displays the bounding boxes of the instances of the level 2 cells in the edited cell (edited cell = root: level 1), as these instances are the objects that can be manipulated. Dali also displays the bounding boxes of the level n+1 cells, which are still unexposed. Note that dali reads cells (*read_cell* command) normally at level 1, but it is possible to specify another default expansion level (see settings menu or ".dalirc" setup file section).

It should be emphasized that the expansion facility is provided only for display purposes: the contents of the instances does not really become part of the edited cell (in the sense of primitive mask geometries). Only the reference to the component cell is part of the edited cell.

1.7.4 Hierarchical Consistency

Care must be taken when a cell has been edited, as other cells higher up in the hierarchy may have to be edited too. That is, the correctness of cells containing the edited cell may be affected by changes in the edited cell, such as a change of its bounding box (see also the command *upd_bbox* in the *inst_menu* for release 3). See also *instcell* (1ICD).

1.8 Layers

The layers or "colors" that are available for the construction of primitive mask geometries and terminals are determined by the *process* one is using. For each project a corresponding process has been specified; see *mkpr* (1ICD).

At startup, dali retrieves the layer information from the project environment. From this information it learns e.g. which layers to use (names), which colors and fill styles to use for their representation, and how these colors should mix when several layers are put on top of each other. From this information dali also learns which layers are the interconnection layers, to be used for terminals. See also *getproc* (1ICD) and *maskdata* (3ICD). In this way, dali can be used for any technology, provided that a proper layer description is present. Note that the fill styles can be changed via the ".dalirc" setup file.

The layers are displayed in the LAYER viewport. In this area the user can select the layers that must be involved in certain operations (e.g. *add_box*, *add_wire*, *add_term*) by toggling the corresponding rectangle.

Via the ".dalirc" setup file (section 1.12) layers may be declared to be *non-editable*. Non-editable layers can not be selected for manipulation of mask geometries or terminals. This is indicated by a white cross in the respective layer rectangles in the LAYER viewport.

At startup, dali assigns colors to the bitplanes of the graphics terminal. These are not only layer colors and their mixtures, but also colors for grid, menu, text, etc. For graphics terminals having only a small number of bitplanes (less than 8), dali has to make a less optimal color assignment, implying that screen updates are sometimes handled less efficiently.

1.9 Panning and Zooming

Dali offers various commands to control which part of the edited cell is displayed in the PICT viewport, i.e. the *window* on the layout. When a cell is read in from the database, it is fit in the viewport, together with a small margin. Zooming-in (*zoom*) and -out (*dezoom*) as well as panning (*center*) can be done with the mouse and cursor: simply point at a new area or new center-point. This "area" zooming mode is default case, but can be changed (see settings menu or ".dalirc" setup file section) to "point" or "fixed" mode. Other commands allow the user to go back to the previous window (*prev*) or to the window that comprises the complete layout (*bbox*). This "*window*"-manipulation commands have '[...]' around them, to distinguish them from other menu commands.

1.10 The Grid

Dali is a *grid* based editor; all coordinates (except for polygons) are rounded to integer coordinates on the *lambda* grid. Absolute coordinate values can be retrieved by means of the *coordnts* command.

Dali provides two user-controlled grids: a *display grid* and a *snap grid*. The display grid is used for display purposes only, as a reference of scale for the designer. With the *disp_grid* command in the *settings* menu the display grid can be switched on or off (*visible* command, default "on"), and a grid spacing can be selected explicitly. The display grid can also be set to *auto-adjust* (default "on"), then dali will select an appropriate grid spacing based on the size of the PICT viewport in terms of lambda's (size of the window). Note that if *auto-adjust* is "on", the user can't select any other grid spacing value. The grid-points are always positioned at coordinates that are 'modulo grid spacing' values. By default, the possible grid spacing values are 1, 2, 4, 8, 10, 20, 50, 100, 1000 and 10000 lambda. The set of possible grid spacing values can be redefined via the ".dalirc" setup file (section 1.12).

The snap grid is used to round coordinates entered by the designer to a multi-lambda grid. The snap grid points to which rounding is performed are displayed as small plus-signs. Note that snapping (rounding to snap grid) is only performed as the snap grid is set *active*, no matter of the snap grid is displayed or not. With the *snap_grid* command in the *settings* menu the snap grid can be switched on or off (*visible* command, default "off"). The snap grid spacing can be selected explicitly and an offset (*offset* command) for the snap grid with respect to the origin can be specified interactively. By default, the

snap grid spacing is 1 lambda and the offset is (0, 0). By default, the possible snap grid spacing values are 1, 2, 4, 8, 10, 20, 50, 100, 1000 and 10000 lambda. The set of possible snap grid spacing values can be redefined via the ".dalirc" setup file (section 1.12).

Note that the display of both grids is controlled by the *grid* command, and that the display grid and/or snap grid are only visible if they are set *visible*. Note that besides these grids dali always displays the axes of its coordinate system (if possible).

1.11 Design Rule Checking

Dali offers facilities to perform on-line design-rule checks. First of all, dali has been interfaced to the *dimcheck* (1ICD) package, to perform complete (multi-layer) checks (*DRC_menu*). When a check has to be performed, dali writes the edited cell from its private workspace to a scratch cell in the database. The dimcheck package performs the actual check on this scratch cell, and dali retrieves the results and removes the scratch cell. It must be noted that by default a *hierarchical* check is performed. That is, the primitives of the edited cell are checked as well as the interaction with and among the instances, but not the complete component cells. This is achieved by running *exp* (1ICD) with the "-h" option before running dimcheck. The default hierarchical expansion mode can be overruled from the ".dalirc" setup file (section 1.12), to have a linear or flat expansion performed for a complete check. Dali offers several commands that can help the user in localizing / displaying the errors that were found (See also chapter 7: DRC_menu).

The second check facility (*check* command in the *box_menu*) is a fast integrated design-rule checker, performing *single-layer* checks: it checks only for design-rule violations caused by mask geometries of the same layer (compare to *autocheck* (1ICD)). The checks are performed only for the area that is specified by the user. All *visible* layout geometries (i.e. primitive mask geometries, terminals, expansion information and sub-terminals) within this area are checked: *What You See Is What You Check*. Errors that are found are displayed automatically.

The integrated checker tries to retrieve its design-rules from a file *dali_drc* (3ICD) that must be present in the process directory of the current process in the standard process library. This file specifies for each layer (field 1) the minimal width (2), the minimal gap (3) for a pair of edges that are both longer than a critical length (5) and the minimal gap (4) if one of the edges is shorter than or equal to this length (5). If this file is not present for the current process a message will be generated.

1.12 The .dalirc Setup File

Dali provides the user with the capability to specify setup data or to execute statements at startup time. To that end dali tries to read commands from a setup file. It first looks for a file named ".dalirc" in the current working directory. If the file does not exist or cannot be opened for reading, dali looks for a file named ".dalirc" in the user's home directory. If that file does not exist or cannot be opened for reading either, dali finally tries for a file named "dalirc" in the process directory. If a setup file is found present and readable, dali executes the commands found in the file.

The commands that can be specified in the setup file are divided into two groups. One group is for specifying setup data, such as the possible snap grid spacing values, the other group contains editing commands that are to be executed by dali, and can be used for purposes of animation.

The syntax and semantics for these commands is given below. A complete command or statement consists of a single line and starts with a (literal) keyword followed by a number of arguments. Other literals are delimited by '"' and '"'. Arguments delimited by '<' and '>' are obligatory (not literal, their value has to be filled in), arguments delimited by '[' and ']' are optional, arguments separated by a '/' means a choice of either one of them. '....' means a variable number of arguments.

1.12.1 Specifying Setup Data

- `default_expan <level>`
  Set default expansion level for the *read_cell* command. Value must lay between 1 to 9. If "no" is specified, then there is always asked for a level.
- `disabled_layers [<v1> <v2> .... <vn>]`
  `v1`, `v2`, ..., `vn` are the names of the layers that are declared to be non-editable. If no layers are specified, then all layers are editable. By default all layers of the given process can be edited.
- `display_grid_values <v1> <v2> .... <vn>`
  `v1`, `v2`, ..., `vn` are the possible values that will be displayed in the display-grid menu. If not specified, a set of default values will be displayed.
- `display_grid_width <aWidth>`
  The program initializes the display grid spacing to `aWidth` (>= 1). This command implies the disabling of the automatic grid adjustment (see section 1.10).
- `dominant ["on"/"off"]`
  The drawing mode can be set to dominant ("on") or transparent ("off"), which is the default mode. No second argument implies "on". In transparent mode mix colors are produced if layers are stacked on top of each other. In dominant mode one layer may hide another layer underneath.

- `drawing_order [<v1> <v2> .... <vn>]`
  This command specifies the drawing order for the layers in case the dominant mode was set. `v1`, `v2`, ..., `vn` are the names of the layers. The order is such that `v1` is the 'top' layer, and hence is drawn last. A new drawing_order command specifies a new drawing order. If no layers are specified, then the default situation is used.

- `drc_options [<options>]`
  Set command-line options for the dimcheck program.

- `exp_options [<options>]`
  Set command-line options for the exp program (see also `flat_expansion`).

- `fill_style <layer> <style#>["+"/"-"<shift>]`
  Set fill style `style#` for certain layer. The following fill styles can be defined: 0=hashed, 1=solid, 2=hollow, 3-5=12.5,25,50% hashed+outline, 6-8=12.5,25,50% hashed (no outline). Note that fill styles 0 and 6 are identical. By the hashed fill styles it is possible to specify a `shift` in the range 1 to 7 (fill lines are shifted that number of pixels).

- `flat_expansion ["on"/"off"]`
  The expansion mode for the *do_check* command in the DRC_menu can be set to linear / flat expansion ("on") or to hierarchical expansion ("off"), which is the default mode. No second argument implies "on".

- `hashed ["on"/"off"]`
  Set the drawing mode for the layers of instances to hashed fill style. No second argument implies "on".

- `image [<image_name>]`
  Define or undefine the instance name for the Sea-of-Gates image. For this instance no bounding boxes, no instance names, no terminal names are displayed. On level 1 the terminals of the image are displayed in hashed style.

- `maxdraw <value>`
  Specifies the maximum number of cell instance repetitions which may be drawn for the Sea-of-Gates image. Default value is 120.

- `no_grid_adjust`
  This command disables the automatic adjustment of the display grid to the size of the window (see section 1.10).

- `set_colornr <colornr> <color>`
  With this command other colors can be assigned to the internal Xdali colortable. However, colornumbers 0 to 7 can not be changed by the user, they are already in use (colors: black, red, green, yellow, blue, magenta, cyan and white). Only colornumbers 8 to 15 can be assigned a new `color` (X Windows colorname or #RGB-value). This command must be at the beginning of the ".dalirc" file.

See also the `set_maskcolornr` command.

- `set_maskcolornr <mask> <colornr>`
  With this command other colors (see `set_colornr`) can be assigned to the masks (layers) of the process in use. This overrules the specifications in the process *maskdata* file. Note that the colornumbers 0 and 15 are in use by the contact hole layers (which are displayed dominant) and that colornumber 15 is default assigned the color 'grey'. This command must be at the beginning of the ".dalirc" file.

- `snap_grid_offset <x_offset> <y_offset>`
  The initial offset of the snap grid with respect to the origin is defined with this command. Default snap grid offset is (0, 0). This command implies the *visible* and *active* setting (see section 1.10).

- `snap_grid_values [v1] [v2] .... [vn]`
  `v1`, `v2`, ..., `vn` are the possible snap grid values that will be displayed in the snap-grid menu. If not specified, a set of default values will be displayed. Initial grid_value is `v1`.

- `snap_grid_width <aWidth>`
  The program initializes the snap grid spacing to `aWidth` (>= 1). Default value is 1 lambda. This command implies the *visible* and *active* setting (see section 1.10).

- `tracker [<mode>]`
  Set tracker mode to "auto", "on" or "off" (see *settings* menu). No second argument implies "on".

- `use_new_name ["on"/"off"]`
  Set the usage of the new_name in place of old_name after the *write_cell* command. No second argument implies "on".

- `via [<via_name>]`
  Define or undefine the first three characters of the instance names for the Sea-of-Gates via cells. Note that this instances are displayed without instance names and terminal names to make them more clear.

- `wire_extension ["on"/"off"]`
  Switch wire extension on or off. No second argument implies "on". See the ADD_WIRE command in the BOX_MENU (section 4.13).

- `wire_width <aWidth>`
  The program initializes the wire-width to `aWidth` (>= 1). If the wire-width is not set from the setup file, dali will ask for a wire-width the first time the *add_wire* command is invoked.

- `wire_width_values <v1> <v2> .... <vn>`
  `v1`, `v2`, ..., `vn` are the values that will be displayed in the wire-width menu. Given values must be >= 1. If not specified, a set of default values will be displayed.

- `zoom_mode <mode>`
  The zooming mode for the menu commands *dezoom* and *zoom* can be set to "area", "fixed" or "point" (see *settings* menu).

1.12.2 Layout Editing Commands / Animation

- `add_val <value>`
  Add integer `value` to internal value.
- `append <xl> <xr> <yb> <yt>`
  Append a rectangle with the specified coordinates to the layout in the selected layers.
- `beep [<volume>]`
  Ring a bell. The `volume` percentage can be +/-100 (default: 0).
- `center <cx> <cy>`
  The new window is centered on (`cx`, `cy`), where `cx` and `cy` are relative to the contents of the dali workspace: (0.0, 0.0) is the lower left corner of the dali workspace and (1.0, 1.0) is the upper right corner of the dali workspace.
- `delete <xl> <xr> <yb> <yt>`
  Delete the layout of the selected layers found in the rectangle of the specified coordinates.
- `expand <level>`
  The hierarchical expansion level is set to `level` (>= 1 and <= 100). The edited cell is expanded with this `level`.
- `goto <name>`
  Jump to the position in the setup file that is labeled `name`. If `name` can not be found in the setup file, then remainder of the ".dalirc" file is skipped.
- `if_val <value> <name>`
  If value is equal to `value` goto position in the setup file labeled `name`.
- `ifnot_val <value> <name>`
  If value is not equal to `value` goto position in the setup file labeled `name`.
- `label <name>`
  Position in the setup file to which can be jumped through a `goto` command.
- `layer <layer> ["on"/"off"]`
  The layer `layer` is either selected ("on") or de-selected ("off") or toggled (no second argument) for editing. Note that `disabled_layers` are always "off".
- `print [<someText>]`
  Print `someText` in the TEXT viewport or `erase` the TEXT viewport.
- `quit`
  Exit immediately the dali program.

- `read <cell_name>`
  Read the cell named `cell_name` into the dali workspace (with default expansion level).
- `redraw`
  Redraw the dali display. Works only after changing 'visible' or 'dominant' mode.
- `set_val <value>`
  Set internal value to integer `value`, which is used by the `if_val` or `ifnot_val` command.
- `sleep <numsec>`
  Wait `numsec` seconds before continuing with the next command.
- `visible <layer> ["on"/"off"]`
  The layer `layer` is either displayed ("on") or not displayed ("off") or toggled (no second argument) for display. Note that `layer` can also be a name like "*bboxes*", "*disp_grid*", "*instances*", "*sub_terms*", "*terminals*", etc. (see *visible* menu).
- `wdw_bbx`
  Redraw the dali display with the full contents of the dali workspace in view.
- `zoom <cx> <cy> <fraction>`
  The size of the new window on the dali workspace is equal to `fraction` times the size of the contents of the dali workspace, and is centered on (`cx`, `cy`), where `cx` and `cy` are relative to the contents of the dali workspace: (0.0, 0.0) is the lower left corner of the dali workspace and (1.0, 1.0) is the upper right corner of the dali workspace.

2. THE MENU STRUCTURE

2.1 The Command Classes

The dali commands can be divided into several classes.

- Basic editing commands to make changes to the edited cell. This class can be divided into three sub-classes.
  1. Commands to handle the primitive mask geometries (representation data).
  2. Commands to add, delete or manipulate terminals (interface data).
  3. Commands to add or delete instances or to change their parameters (hierarchical data).
- Database interaction / workspace commands which allow the user to read cells from the database into the private workspace of dali, store them again into the database when he thinks he has made some valuable changes, and a command to erase the workspace.
- Additional support commands which allow the user to change his window, set the visibility of some sort of objects or to put a grid over the design.
- Commands to check if the layout that has been generated obeys the design-rules posed upon them.

2.2 The Main Menu

Dali has a hierarchical menu structure, based on the command classes recognized above. Upon starting, the *main* menu is shown on the screen, from which the user may select one of the sub-menu's (command clusters). Upon doing so, the set of commands belonging to the cluster is shown, and the user may select one of them. The bottom command in each sub-menu allows the user to return to the main menu. The sub-menu's and commands in the main menu are:

- **quit**
  With this command one can quit from the editor. After a confirmation has been given the edit session is terminated. The contents of the workspace will *not* be saved on this occasion. If a valuable layout description is present in the workspace, it should be saved explicitly in the database by means of the *write_cell* command, before terminating the edit session.
- **visible**
  In this sub-menu the user can set layers as well as other graphical information visible / not-visible on the screen. This menu is explained in more detail in chapter 9.

- **DB_menu**
  Commands to interact with the *data manager* (database). Also commands to expand the instance(s) for display, as well as commands to erase the workspace. The commands in this menu are explained in more detail in chapter 3.
- **box_menu**
  Commands to manipulate the *primitive mask geometries* of the cell being edited. This includes the manipulation of polygons containing 45 degree edges, buffer operations, wire editing and a fast integrated design-rule checker performing single-layer checks. The commands in this menu are explained in more detail in chapter 4.
- **term_menu**
  Commands to manipulate the *terminals*, including the manipulation of a terminal buffer as well as the possibility to "lift" terminals of instances. The commands in this menu are explained in more detail in chapter 5.
- **inst_menu**
  Commands to manipulate *instances*, including the manipulation of arrays of instances and the possibility to add instances of cells that have previously been imported from other projects. The commands in this menu are explained in more detail in chapter 6.
- **DRC_menu**
  Contains the commands to do *design-rule checking* and to show the results on the screen. Dali contacts the *dimcheck* (1ICD) package to perform the actual check. The commands in this menu are explained in more detail in chapter 7.
- **info_menu**
  In this sub-menu some *information* commands are gathered. For instance, commands to get information about the active cell and the process which is used. The commands in this menu are explained in more detail in chapter 8.
- **settings**
  This sub-menu contains the commands for setting *dali* working modes. For instance, commands to control the display grid and snap grid and to set the tracker mode. The commands in this menu are explained in more detail in chapter 9.

Some of the support commands appear in more than one sub-menu. In particular the window operations as well as the *grid* and *coordnts* command are present in several sub-menu's for convenience' sake.

2.3 Command Selection / Operation

Whenever a command is selected the corresponding area is highlighted and remains highlighted until the operation has been finished or another command has been selected. Certain commands are *self-repeating*; when they finish they are automatically re-selected. The menu item remains highlighted, until another command is selected. This

has the advantage that these commands can be applied repeatedly without being interrupted by an explicit re-selection by the user. Examples are the *zoom*, *coordnts* and *add_box* commands. Note that *self-repeating* commands can be de-selected by selecting them again, this must be done before a keyboard command can be given.

Certain commands require additional information to be specified during their execution. For instance, coordinates, names, terminals, instances or modes of operation may have to be specified. When alphanumerical information has to be entered dali prompts for this in the TEXT viewport. The 'Return'-key finishes the keyboard input.
The 'Delete'- or 'BackSpace'-key can be used to correct typing mistakes.

When the user has to select an item from a set of alternatives dali may present a menu for this in the MENU viewport, from which the user has to make a choice. When coordinates have to be specified in the PICT viewport, it is often allowed to point at a command or layer instead. As we already mentioned in section 1.3, this gives the user the opportunity to cancel the command in progress by selecting another command, or to (de)select the layers on which the command will operate.

2.3.1 Keyboard Commands

Some commands can be activated with keyboard keys, this is only possible in the *main* menu and the sub-menu's of the *main* menu (and some other sub-sub-menu's).
At this moment the following keys are defined:

— 'd': Do expansion with default level.

— 'e': Do expansion with *all_exp* command.

— 'i': Do individual expansion (*indiv_exp* command).

— 'v': Display visible menu.

— 'q': Exit immediately dali, sure quit!

— 'Esc': Exit (escape) current menu, to previous menu.

3. The DB_MENU

This menu contains the following commands to interact with the data manager (database), as well as some related commands:

1. RETURN: Return to the main menu.

   After selection of this command the DB_menu will be left and the *main* menu will be shown again.

2. READ_CELL: Read a cell from the database.

   Dali presents the list of local cells from which one cell can be selected. Note that only the local cells are presented and not also the alias names of imported remote cells. If this is a long list only part of it is presented at a time, and -*prev*-, -*next*- and -*keyboard*- menu items are included to allow the user to move back and forth through the list. The -*keyboard*- menu item can be used to give interactively the cell_name, but it can also be used to search for a cell_name beginning with certain character (and it sets a new list position). If a search is requested, the character must be followed by an asterisk '*'. Note that the request "**" sets the list back to the begin position. At last there is one special request: If only one asterisk "*" is given, then the cell list is set back to its previous position. A cancel possibility is included as the last menu item. Upon selection of a cell dali asks for a confirmation if there is already a cell present in the workspace. If confirmed, the workspace is cleared. The layout data of the selected cell is loaded into the workspace and the picture appears. The objects that were present in the geometry and terminal buffer will still exist after the execution of this command.

3. WRITE_CELL: Write a cell to the database.

   With this command all the information pertaining to the edited cell that is present in the workspace, can be stored in the database. One has the option to give the cell a new name or to store it with its present name if it already has one. A cancel possibility is included as the bottom sub-menu item. Note that by *new name* mode the cell CREATE-mode is used, otherwise the cell UPDATE-mode. If the edited cell is stored under a new name while it already has one, then the old name will still be related to the workspace copy that remains after the *write_cell* command has finished. This permits a quick *save* of an intermediate result under a different name. However, if the *new mode* setting (in settings menu) or the *use_new_name* command (".dalirc" setup file) is activated, then the new name is used instead of the old name for the edited cell. If the edited cell has been built from scratch and has not been saved before, i.e. no name is yet related to the workspace copy, then the new name will be related to the workspace copy after the *write_cell* command has finished.

   If a design rule check has been performed with the *do_check* command (DRC_menu) on the cell that is being saved, the results of this check are present in

the file "chk_mod.ck" in the project directory. *Write_cell* changes the name of this file to "cell_name.ck", where cell_name stands for the name of the cell that is written to the database. This file can then be used during a next session with the same cell (see also chapter 7 on the DRC_menu).

4. ERASE_ALL: Erase the workspace.

   After a confirmation has been given, the workspace is cleared. The geometry and terminal buffer will remain intact.

5. INDIV_EXP: Expand an individual instance.

   Upon the selection of this command one first has to identify the instance one wants to expand. This can be done by just pointing at a position within the bounding box of the instance (repetitions included). See also section 1.7.2. After a unique instance has been identified a sub-menu appears from which one may choose the level of the expansion. The current level of the selected instance is initially highlighted. If the selected level is the same as the current level, the command is canceled. Otherwise, the selected instance is (un)expanded to the selected level. Thus, if the level is increased, additional detail will be shown on the screen for this instance. If the level is decreased, the instance will be erased and drawn again with less detail in it.

6. ALL_EXP: Expand all instances.

   Upon the selection of this command a sub-menu appears from which one may choose the level of the expansion. The current global expansion level is initially highlighted. Note that this current level can be the level, which is set with the *expand* command from the ".dalirc" setup file. If the selected level is the same as the current level, the command is canceled. Otherwise, all instances are (un)expanded to the selected level. That is, if the level has to be increased for a certain instance because the new global level is larger than its current level, additional detail will be shown on the screen for this instance. If the level is decreased, the instance will be erased and drawn again with less detail in it. Note that level 10 can be any selected level in the range 10 to 99. The maximum possible level (= 100) can directly be chosen with the *maximum* command. At last, with the *keyboard* command can interactively every level be chosen in the range 1 to 100.

4. The BOX_MENU

This menu contains the commands for the manipulation of the primitive mask geometries.

1. RETURN: Return to the main menu.

   After selection of this command the box_menu will be left and the main menu will be shown again.

2. GRID: Toggle grids.

   Via this command the display grid and the snap grid can be displayed according to the settings as controlled via the *disp_grid* and *snap_grid* commands (settings menu). A grid is displayed only if it has been switched on. By subsequently selecting the *grid* command the display of the grids is alternately turned on and off: toggle grids. If the display of the grids is turned on, the display grid spacing is reported in the TEXT viewport.

3. BBOX: Set the window to the bounding box of the edited cell.

   The new window is the smallest window in which the bounding box of the design, together with a small margin, fits.

4. PREV: Set the window to the previous window.

   The new window is the window that was displayed just before the current one.

5. CENTER: Set the window with a new center but at the same scale.

   By fixing one point in the PICT viewport, the new center of the window is specified. The picture is redrawn at the same scale but with the specified point as the new center. Thus, using the *center* command one can *pan* with variable step and direction. This command is self-repeating.

6. ZOOM: Zoom in with the help of a rubber-box cursor.

   By fixing two points with the help of a rubber-box cursor a new, smaller, window can be specified. The new window is the smallest window of a given ratio, in which the specified part of the design fits. This command is self-repeating.

7. DEZOOM: Fit the current window in the cursor area.

   An area has to be specified with the help of a rubber-box cursor. The window is then set in such a way, that the current window fits in the specified area. As a consequence, a larger part of the edited cell becomes visible. This command is self-repeating.

8. COORDNTS: Show the coordinates of a certain point.

After the selection of this command one may point at a certain position in the PICT viewport. A small cross subsequently appears at the closest grid point and the coordinates corresponding to this grid point are reported in the TEXT viewport. This command is self-repeating; the coordinates of multiple grid points can be retrieved without interruption.

9. ADD_BOX: Add a box.

   After the selection of this command one can add boxes by just fixing two points. After the first point has been fixed, a rubber-box cursor supports the positioning of the second point in an easy way. The command operates only on the activated layers. Before a second point is given any layer may be (de)activated. Note that this is only true for visible and editable layers.

   The *add_poly* command is self-repeating; several boxes can be added without interruption until one selects another command or de-selects the command.

10. DEL_BOX: Delete a box.

    This command operates in the same way as *add_box* does, except that now the mask geometries for the activated layer(s) are *deleted* inside the specified area (box).

    After the execution of the command the deleted primitive layout will still be present in the geometry buffer. This implies that one can recover from non-desirable results using the *put* command. Also a *move* can be made by deleting mask geometries at one spot and *put* them somewhere else.

11. ADD_POLY: Add a polygon.

    After the selection of this command a menu is presented from which one can add polygons by subsequently entering the corner points of the contour polygon. If the 45 degree mode has been switched on (the default), non-orthogonal mask geometries can also be added. After the first point has been fixed a rubber-line cursor assists the positioning of the other corner points. Upon entering such a point the closest orthogonal or 45 degree line segment is generated by either adjusting the x- or y-coordinate of the point that has been entered. Note that with the *use_big* setting the opposite adjustment takes place. With the *set_cross* command a big cross is displayed on the position of the first point. A polygon may consist of up to 127 line segments. Incorrect line segments can be corrected by just *walking back* over the parts that have already been entered or by the use of the *walk_back* command. A polygon may be self-intersecting. Upon closing the polygon (last point = first point), the corresponding mask geometries are added to the internal workspace and painted on the screen. Closing of the polygon can be forced with the *next* command. The polygon can be cancelled (if not yet closed) with the *cancel* command. With the *return* command the ADD_POLY menu is exited.

The *add_poly* command is also self-repeating like the *add_box* command.

12. DEL_POLY: Delete a polygon.

    This command operates in the same way as *add_poly* does, except that now the mask geometries for the activated layer(s) are *deleted* inside the specified contour polygon.

    After the execution of the command the deleted primitive layout will still be present in the geometry buffer. Like the *del_box* command.

13. ADD_WIRE: Add a wire, using a rubber-line cursor.

    A menu is presented from which the wiring process can be controlled. It includes commands to specify a wire width, to manipulate the display window, and to start and finish wire-editing.

    If no wire width has been specified previously or from the ".dalirc" setup file, dali first asks for a wire width. See the command *set_width* below. A wire is entered by specifying its center-line in the PICT viewport. After the first point has been fixed, a rubber-line cursor assists the positioning of the other points. Upon entering such a point the closest orthogonal or 45 degree line segment is fixed by either adjusting the x- or y-coordinate of the point that has been entered. 45 degree segments can be entered only if the *45 degree* mode has previously been switched on. The center-line may consist of up to 127 line segments. Incorrect line segments can be corrected by just *walking back* over the parts that have already been entered. A wire of the specified width is generated for the selected layers if the *next* or *ready* command is selected, or if the same wire-point is entered twice. The wire is generated either with or without an extension of half the wire width for the first and last wire segment, as controlled by the *extension* command. Default is no extension.

    The wire-menu contains the following commands:

    — *ready*: A wire is generated for the center-line that has been entered so far. The corresponding mask geometries are added to the edited cell and appear on the screen. This finishes the *add_wire* command and a return is made to the box_menu.

    — *next*: A wire is generated for the center-line that has been entered so far. The corresponding mask geometries are added to the edited cell and appear on the screen. As opposed to *ready* this does not finish the *add_wire* command. A new center-line for the *next* wire, having the same width, can be entered right away.

    — *cancel*: The center-line that has been entered so far is erased and one may start again to enter a new center-line. No changes are made to the edited cell.

— *grid*: Toggle grids. See description of this command above.

— *bbox*: Set the window to the bounding box of the edited cell. See description of the *bbox* command above. This command can be intermixed with the specification of center-line segments, thereby allowing the user to view the complete layout while specifying the wire.

— *prev*: Set the window to the previous window. See description of the *prev* command above. This command can be intermixed with the specification of center-line segments.

— *center*: After this menu item has been selected, one can enter a new center for the window, just as with the *center* command described above. As this can be intermixed with the specification of center-line segments, it allows the user to move over the layout while specifying the wire. This increased flexibility permits longer wires to be added, while viewing the layout at a reasonable scale.

— *zoom*: After this menu item has been selected, one can specify a smaller window, just as with the *zoom* command described above. As this can be intermixed with the specification of center-line segments, it allows the user to focus on a small part of the layout when fixing e.g. a start- or end-point of the wire.

— *dezoom*: After this menu item has been selected, one can specify a larger window, just as with the *dezoom* command described above. As this can be intermixed with the specification of center-line segments, it allows the user to display a larger part of the layout, e.g. after fixing a point at a more detailed level. By subsequently zooming-in on another part of the layout long wires can comfortably be entered with reasonable accuracy.

— *set_width*: As shown in a sub-menu, a wire width can be specified either by selecting a predefined value, by entering a width via the keyboard (*keyboard* command) or by interactively pointing at two points in the PICT viewport (*interact* command). In the last case, the selected width is the maximum distance in either the x-direction or the y-direction between the two points that have been entered. A wire width can not be specified while a center-line is being entered. If the snap grid is on only even wire widths are permitted.

— *extension*: As can be controlled with this command, a wire can be generated either with or without an extension for the first and last wire segment. If extension is switched off the generated wires will start and stop at the first and last point of the center-line that has been entered (with rounding to grid for odd wire widths). If extension is switched on an overlap of half the wire width will be generated around the first and last point of the center-line. This is particularly convenient when editing wires on a course snap grid to connect, for

instance, contacts positioned on this snap grid. Default is no extension, but extension can be switched on via the ".dalirc" setup file (section 1.12).

— *45 degree*: With this command the 45 degree mode can be switched on or off. This mode can only be switched, before the first point of the center-line has been entered.

14. YANK: Fill the mask geometry buffer, using a rubber-box cursor.

    A rectangular area has to be specified with the help of a rubber-box cursor. Primitive mask geometries of the edited cell that lie within the specified area are copied into the geometry buffer. The *yank* command operates only on the activated layers.

15. PUT: Add the contents of the geometry buffer to the edited cell.

    After the selection of this command one can position the buffer contents by specifying a position for the lower left corner of its bounding box in the PICT viewport. While an impression of the buffer contents is being displayed new positions can repeatedly be specified. The *put* command can be finished by selecting the *return* menu item in the sub-menu that is presented. By selecting the *put_buf* menu item on the other hand, the contents of the geometry buffer are added to the edited cell at the specified position. This is also performed if the same cursor position is entered twice. Note that only the buffer contents for the visible layers is put back. The cursor position related to the contents of the buffer can be changed with the *set_cursor* menu item. In the PICT viewport this position is marked with a cross. At last, the buffer contents can also be mirrored and rotated around this cross position.

16. CHECK: Perform a single-layer design-rule check.

    A rectangular area has to be specified with the help of a rubber-box cursor, which is subsequently checked for design rule violations. All *visible* layout geometries (i.e. primitive mask geometries, terminals, expansion information and sub-terminals) within the specified area are checked: What You See Is What You Check. Only *single-layer* checks are performed (width and gap). That is, it doesn't check for design-rule violations caused by a combination of mask geometries from two or more different layers (compare to *autocheck* (1ICD)). The violations that are found are displayed on the screen. No check is performed if dali has not been able to retrieve the design-rules at the start of the session (section 1.11). In this case, a message is generated.

5. The TERM_MENU

This menu contains the commands for the manipulation of the terminals of the edited cell.

1. RETURN: Return to the main menu.

   After selection of this command the term_menu will be left and the main menu will be shown again.

2. GRID: Toggle grids.

   Via this command the display grid and the snap grid can be displayed according to the settings as controlled via the *disp_grid* and *snap_grid* commands (settings menu). A grid is displayed only if it has been switched on. By subsequently selecting the *grid* command the display of the grids is alternately turned on and off: toggle grids. If the display of the grids is turned on, the display grid spacing is reported in the TEXT viewport.

3. BBOX: Set the window to the bounding box of the edited cell.

   The new window is the smallest window in which the bounding box of the design, together with a small margin, fits.

4. PREV: Set the window to the previous window.

   The new window is the window that was displayed just before the current one.

5. CENTER: Set the window with a new center but at the same scale.

   By fixing one point in the PICT viewport, the new center of the window is specified. The picture is redrawn at the same scale but with the specified point as the new center. Thus, using the *center* command one can *pan* with variable step and direction. This command is self-repeating.

6. ZOOM: Zoom in with the help of a rubber-box cursor.

   By fixing two points with the help of a rubber-box cursor a new, smaller, window can be specified. The new window is the smallest window of a given ratio, in which the specified part of the design fits. This command is self-repeating.

7. DEZOOM: Fit the current window in the cursor area.

   An area has to be specified with the help of a rubber-box cursor. The window is then set in such a way, that the current window fits in the specified area. As a consequence, a larger part of the edited cell becomes visible. This command is self-repeating.

8. ADD_TERM: Add a terminal, using a rubber-box cursor.

This command operates in the same way as *add_box* does, but now dali also asks for a terminal name, which has to be entered via the keyboard. Only names that have not yet been used for other terminals in the edited cell are allowed. This command operates only on the activated *interconnection* layers. For each of these layers dali prompts for a name and adds a terminal. The *add_term* command is self-repeating.

9. DEL_TERM: Delete a terminal.

   The terminal that has to be deleted must be identified explicitly by pointing at it in the PICT viewport. This terminal selection procedure is described in more detail in section 1.6. After a unique terminal has been identified it is deleted. The name of the deleted terminal is reported in the TEXT viewport. The *del_term* command is self-repeating.

10. DEL_AREA: Delete all terminals within a specified area.

    After the selection of this command a rectangular area has to be specified with the help of a rubber-box cursor. All terminals (arrays of terminals) of editable layers that lie completely within this area are deleted. That is, a terminal that is partly situated outside the specified area is left in its present state. The *del_area* command does not take the activated layers into account.

    All deleted terminals are saved in the terminal buffer. Thus, one can *undo* the deletion by placing this buffer at the old position or one can *move* the terminals to another position by means of the command sequence *del_area* and *put_buf*.

11. YANK_AREA: Yank all terminals within a specified area.

    This command operates in the same way as *del_area* does, but in place of deleting, the terminals are copied to the terminal buffer.

12. PUT_BUF: Add the contents of the terminal buffer to the edited cell.

    One can position the terminal buffer contents by specifying a position for the lower left corner of its bounding box, just as with the *put* command in the box_menu. For the operation of this command see the *put* command. Note that the *put_buf* command only works if the terminals are visible and not takes in account the visibility of the layers.

13. MOV_TERM: Move a terminal to a new position.

    The terminal that has to be moved must be selected explicitly in the PICT viewport. If a terminal is selected, then a new position may be entered by pointing at a certain position in the PICT viewport. This specifies the new position for the lower left corner of the terminal. If the terminal has been *arrayed*, the lower left corner of the (0,0) occurrence is taken as the reference point. In this case the complete array is moved. The *mov_term* command is self-repeating.

14. ARRAY_TERM: Change the array parameters of a selected terminal.

    The array parameters are the number of repetitions in both directions: *nx* and *ny*, and the repetition spacing in both directions: *dx* and *dy*. The default value for nx and ny is 0. The repetition spacing in the x-direction, dx, is defined as the distance between the left side of the (0,k) occurrence and the left side of the (1,k) occurrence. In the same way, dy is defined as the distance between the bottom side of the (k,0) occurrence and the bottom side of the (k,1) occurrence. Default values for dx and dy are the width and height of the individual occurrence: abutment.

    A sub-menu is presented for the selection of one of the array parameters. A cancel possibility is included as the bottom menu item. If nx or ny is selected, a new value can be entered via the keyboard. A negative nx or ny will cancel the command. If dx or dy is selected, a new spacing can be specified by pointing at a position in the PICT viewport. In case of dx, the x-coordinate of the specified point is taken as the new left side of the (1,k) occurrence(s). In case of dy, the y-coordinate is taken as the new bottom side of the (k,1) occurrence(s).

15. LIFT_TMS: Create terminals on top of the terminals of an instance.

    First the user has to select an instance by pointing at it in the PICT viewport. See also section 1.7.2. Then he has to specify at which side(s) of the bounding box of this instance he would like to create terminals. This can be done by toggling the corresponding side-specifiers (*top, bottom, right, left*) in a sub-menu until the *ready* menu item is selected. After that, another sub-menu is presented from which the user has to choose how he would like to assign the names to the new terminals: use the names of the underlaying terminals (*old name*), give an extension to the old names (*extension*) or give totally new names (*new name*). A cancel possibility is included as the bottom menu item.

    When all the necessary information has been supplied, dali starts with the creation of the terminals. Only the terminals of the selected instance that lie exactly on the specified side(s) of the bounding box are lifted. This may be interrupted if the name that has been generated for a new terminal is already associated with an existing terminal (which the *lift_tms* command itself might just have created). On this occasion dali prompts for a new name, which has to be entered via the keyboard. After the complete process has been finished the generated terminals become visible (note: if the terminals are visible). Note that the terminals of non-editable layers are skipped by the *lift_tms* command.

6. The INST_MENU

This menu contains commands for manipulation of instances.

1. RETURN: Return to the main menu.

   After selection of this command the inst_menu will be left and the main menu will be shown again.

2. GRID: Toggle grids.

   Via this command the display grid and the snap grid can be displayed according to the settings as controlled via the *disp_grid* and *snap_grid* commands (settings menu). A grid is displayed only if it has been switched on. By subsequently selecting the *grid* command the display of the grids is alternately turned on and off: toggle grids. If the display of the grids is turned on, the display grid spacing is reported in the TEXT viewport.

3. BBOX: Set the window to the bounding box of the edited cell.

   The new window is the smallest window in which the bounding box of the design, together with a small margin, fits.

4. PREV: Set the window to the previous window.

   The new window is the window that was displayed just before the current one.

5. CENTER: Set the window with a new center but at the same scale.

   By fixing one point in the PICT viewport, the new center of the window is specified. The picture is redrawn at the same scale but with the specified point as the new center. Thus, using the *center* command one can *pan* with variable step and direction. This command is self-repeating.

6. ZOOM: Zoom in with the help of a rubber-box cursor.

   By fixing two points with the help of a rubber-box cursor a new, smaller, window can be specified. The new window is the smallest window of a given ratio, in which the specified part of the design fits. This command is self-repeating.

7. DEZOOM: Fit the current window in the cursor area.

   An area has to be specified with the help of a rubber-box cursor. The window is then set in such a way, that the current window fits in the specified area. As a consequence, a larger part of the edited cell becomes visible. This command is self-repeating.

8. ADD_INST: Add instance of local / imported cell to the edited cell.

   After selecting this command one must first make a choice between the *local* and the *imported* cells.

After the choice, if present, dali displays this list of cells in the MENU viewport. If this is a long list only part of it is presented at a time, and -*prev*- and -*next*- menu items are included to allow the user to move back and forth through the list. One can either select a cell_name or cancel the command by selecting the last menu item. A selected cell can be placed in the edited cell by fixing its lower left corner: by pointing at a certain position in the PICT viewport the placement of the lower left corner of the selected component cell is specified and its instance will appear there. The expansion information of the new instance will be displayed right away, up to the global expansion level (set by *all_exp*).

With the *keyboard* menu item the cells can interactively be selected. It is also possible to search for a cell_name (see the *read_cell* command in the DB_menu).

9. DEL_INST: Delete an instance.

   An instance has to be identified by pointing at a position within its bounding box (repetitions included). See also section 1.7.2. The instance that is identified is deleted. The name of the component cell of the deleted instance is reported in the TEXT viewport. Explicit specification is required in this case to prevent annoying mistakes. The *del_inst* command is self-repeating; multiple instances can be deleted without having to reselect the command.

10. MOV_INST: Move an instance.

    By pointing at a certain position in the PICT viewport the new position for the lower left corner of the selected instance is specified. If the instance has been *arrayed*, the lower left corner of the (0,0) occurrence is taken as the reference point. In this case the complete array is moved. The *mov_inst* command is also self-repeating.

11. MIR_INST: Mirror an instance.

    If an instance is selected, then dali presents a sub-menu by means of which the user has to specify whether the mirroring should be performed around the x-axis (*x*) or around the y-axis (*y*). A cancel possibility is included as the bottom menu item. If an axis is selected, the instance is mirrored around this axis in such a way that its lower left corner remains at its original position. If the instance has been *arrayed*, all individual occurrences will be mirrored.

12. ROT_INST: Rotate an instance.

    The selected instance is instantly rotated 90 degrees counter-clockwise. The lower left corner of the instance remains at its original position. If the instance has been *arrayed*, all individual occurrences will be rotated.

13. ARRAY_INST: Change the array parameters of an instance.

The array parameters are the number of repetitions in both directions: *nx* and *ny*, and the repetition spacing in both directions: *dx* and *dy*. The default value for nx and ny is 0. The repetition spacing in the x-direction, dx, is defined as the distance between the left side of the (0,k) occurrence and the left side of the (1,k) occurrence. In the same way, dy is defined as the distance between the bottom side of the (k,0) occurrence and the bottom side of the (k,1) occurrence. Default values for dx and dy are the width and height of the individual occurrence: abutment.

After the selecting an instance, a sub-menu is presented for the selection of one of the array parameters. A cancel possibility is included as the bottom menu item. If nx or ny is selected, a new value can be entered via the keyboard. A negative nx or ny will cancel the command. If dx or dy is selected, a new spacing can be specified by pointing at a position in the PICT viewport. In case of dx, the x-coordinate of the specified point is taken as the new left side of the (1,k) occurrence(s). In case of dy, the y-coordinate is taken as the new bottom side of the (k,1) occurrence(s).

14. NAME_INST: Name / rename an instance.

    The instance name of the selected instance of the component cell can be set or renamed. The current instance name and cell_name are reported in the TEXT viewport and there is asked for a new name. The new name must be entered via the keyboard. This command is also self-repeating.

15. SHOW_INST: Show the name of an instance.

    The instance name and cell_name of the selected instance are reported in the TEXT viewport. This command is also self-repeating.

16. UPD_BBOX: Update the bounding box of an instance (release 3 only).

    If one has edited a cell that is instantiated in the edited cell, dali may no longer display a correct bounding box for the instance(s) of the modified component cell. By selecting such an instance and performing the *upd_bbox* command, the bounding box can be corrected.

    In order to maintain data consistency with respect to the bounding boxes one has to apply this command to all the 'father' cells, if a 'son' cell has been edited in such a way that its bounding box has changed.

7. The DRC_MENU

In this menu an interactive interface to the *dimcheck* (1ICD) package is provided. It contains the following commands to perform design rule checks and show their results:

1. RETURN: Return to the main menu.

   After selection of this command the DRC_menu will be left and the main menu will be shown again.

2. GRID: Toggle grids.

   Via this command the display grid and the snap grid can be displayed according to the settings as controlled via the *disp_grid* and *snap_grid* commands (settings menu). A grid is displayed only if it has been switched on. By subsequently selecting the *grid* command the display of the grids is alternately turned on and off: toggle grids. If the display of the grids is turned on, the display grid spacing is reported in the TEXT viewport.

3. BBOX: Set the window to the bounding box of the edited cell.

   The new window is the smallest window in which the bounding box of the design, together with a small margin, fits.

4. PREV: Set the window to the previous window.

   The new window is the window that was displayed just before the current one.

5. CENTER: Set the window with a new center but at the same scale.

   By fixing one point in the PICT viewport, the new center of the window is specified. The picture is redrawn at the same scale but with the specified point as the new center. Thus, using the *center* command one can *pan* with variable step and direction. This command is self-repeating.

6. ZOOM: Zoom in with the help of a rubber-box cursor.

   By fixing two points with the help of a rubber-box cursor a new, smaller, window can be specified. The new window is the smallest window of a given ratio, in which the specified part of the design fits. This command is self-repeating.

7. DEZOOM: Fit the current window in the cursor area.

   An area has to be specified with the help of a rubber-box cursor. The window is then set in such a way, that the current window fits in the specified area. As a consequence, a larger part of the edited cell becomes visible. This command is self-repeating.

8. COORDNTS: Show the coordinates of a certain point.

After the selection of this command one may point at a certain position in the PICT viewport. A small cross subsequently appears at the closest grid point and the coordinates corresponding to this grid point are reported in the TEXT viewport. This command is self-repeating; the coordinates of multiple grid points can be retrieved without interruption.

9. IND_ERR: Show error-data of an error pointed at.

   Upon selection of this command one may point at an error in the PICT viewport. The errors can be displayed as white rectangles by e.g. *error*, *chk_file* or *do_check*. The details of the error pointed at are displayed in the TEXT viewport. Information displayed is e.g. the design rule that was not obeyed and the coordinates where the error occurred. This command is self-repeating; one may point at the next error without interruption.

10. NXT_IN_W: Display the next error from the error list inside the current window.

    Using the *nxt_in_w* command, one may walk along the internal list of design rule errors that has been built with the *chk_file* or *do_check* command. Upon selection of this command the next error from the list that lies *inside* the current window is displayed as a white rectangle in the PICT viewport. More detailed textual information about the error is presented in the TEXT_window. The list is traversed in a wrap-around fashion; when the end of the list is reached, dali continues with the first list item.

11. NXT_ERR: Display the next error from the error list.

    Using the *nxt_err* command, one may walk along the internal list of design rule errors that has been built with the *chk_file* or *do_check* command. Upon selection of this command the next error from the list is displayed as a white rectangle in the PICT viewport and more detailed textual information is presented in the TEXT_window. If the next error does not fall within the current window, dali automatically moves the window at the same scale (*panning*, compare *center*) such that the error will be located in the center of the window. The list is traversed in a wrap-around fashion; when the end of the list is reached, dali continues with the first list item.

12. ERROR: Toggle the visibility of the current error list.

    By subsequently selecting this command the visibility of the errors from the list of design rule errors, which may be built with the *chk_file* or *do_check* command, is alternately turned on and off. If the visibility is turned on, all errors from the list are displayed in the PICT viewport as white rectangles. If either all errors or a single error were being displayed, selection of the *error* command will turn off their visibility.

13. CHK_FILE: Load the errors from a previously generated error file.

    Upon the selection of this command dali tries to read a file named "cell_name.ck" from the project directory, where cell_name stands for the name of the cell one is editing. This file contains the results of a previously performed dimcheck upon the cell, before it was called for the edit session. The errors are placed in the internal error list and are displayed in the PICT viewport.

14. DO_CHECK: Perform a design rule check on the edited cell.

    A choice can be made out of the following sub-commands:

    — *check*: This command starts the requested design rule check. To do this, the edited cell that is present in the workspace of dali is saved in the database as a temporary cell with the name "chk_mod". This cell is subsequently expanded by the program *exp* (1ICD) and checked by the program *dimcheck* (1ICD). By default, expansion is performed *hierarchically* (*exp* is run with the option "-h"). Via the ".dalirc" setup file (section 1.12) this can be switched to linear / flat expansion. The results of *dimcheck* are placed in the file "chk_mod.ck" in the project directory. This file is subsequently read by dali, an internal error list is built and the errors are displayed in the PICT viewport. The temporary cell "chk_mod" is removed from the database, after the check has been performed.

    — *cancel*: This sub-command cancels the DO_CHECK command.

    — *linear*: Selects linear expansion mode in place of hierarchical.

    — *hierarch*: Selects hierarchical expansion mode in place of linear.

    — *set_opt*: This sub-command makes it possible to set command-line options for either the program *dimcheck* or *exp*.

    — *show_opt*: With this sub-command it is possible to show the used command-line options for either the program *dimcheck* or *exp*.

8. The INFO_MENU

This menu contains the following information commands

1. RETURN: Return to the main menu.

   After selection of this command the info_menu will be left and the main menu will be shown again.

2. WINDOW: Present information about the current window.

   The current window settings are reported in the TEXT viewport. This includes the coordinates of the piece of layout that is currently being displayed in the PICT viewport.

3. PROCESS: Present information about the process of the current project.

   Some information about the process that corresponds to the project in which dali is running, is reported in the TEXT viewport. This includes the name of the process and the type process and the value of lambda in microns (micrometer) that has been set for this project.

4. CELL: Present information about the edited cell.

   Some information about the edited cell in the workspace is reported in the TEXT viewport. This includes the name of the cell and its bounding box and its current level of expansion.

5. S-O-GATES: Present information about Sea-of-Gates items.

   There is information about the following Sea-of-Gates items:

   — *image_name*: Displays the instance name which is used for the Sea-of-Gates 'image' cell.

   — *via_name*: Displays the first three characters of instance names which are used for the Sea-of-Gates 'via' cells.

   — *maxdraw*: Displays the value (default: 120) of the maximum number of repetitions of the Sea-of-Gates 'image' which are drawn.

9. The SETTINGS MENU

This menu contains the following settings commands

1. RETURN: Return to the main menu.

   After selection of this command the settings menu will be left and the main menu will be shown again.

2. GRID: Toggle grids.

   Via this command the display grid and the snap grid can be displayed according to the settings as controlled via the *disp_grid* and *snap_grid* commands (settings menu). A grid is displayed only if it has been switched on. By subsequently selecting the *grid* command the display of the grids is alternately turned on and off: toggle grids. If the display of the grids is turned on, the display grid spacing is reported in the TEXT viewport.

3. DISP_GRID: Control display grid settings

   A sub-menu is presented from which the settings for the display grid can be controlled. The keyboard commands (see section 2.3) operates also in this menu. The visibility of the grid can be switched on and off via the *visible* menu item. The grid spacing can be controlled either by explicitly selecting a value from the possible grid spacing values displayed in the menu, or by setting the display grid to *auto-adjust*. In the latter case dali will select an appropriate grid spacing, based on the size of the PICT viewport in terms of lambda's (size of the window). Note that in *auto-adjust* mode the user can not change the grid spacing values.

4. SNAP_GRID: Control snap grid settings

   A sub-menu is presented from which the settings for the snap grid can be controlled. The keyboard commands (see section 2.3) operates also in this menu. The visibility of the snap grid can be switched on and off via the *visible* menu item. The snap grid spacing can be controlled by explicitly selecting a value from the possible grid spacing values displayed in the menu. An offset with respect to the origin can be specified interactively via the *offset* menu item. By fixing a point in the PICT viewport, the snap grid is positioned such that the newly entered point is one of the snap grid points. Of actual snapping is performed, i.e. snap grid is active, depends on the setting of the *active* menu item.

5. ZOOM MODE: Set zooming mode area / point / fixed.

   The zooming mode can be set to area (default), point or fixed. With area mode one must specify the zooming area, with point mode only a point must be specified and with fixed mode nothing need to be specified with the cursor (centre of the PICT viewport is used). By fixed and point mode the zooming factor is always two. The zoom mode can also be set via the ".dalirc" setup file (section 1.12).

6. NEW MODE: Set use of old / new cell_name after write.

   Normally after the *write_cell* command (in the DB_menu), if you write the cell under a new name, the old name remains in use for the cell in the workspace which can be edit again. In new mode however the new name is used for the edited cell. The new mode can also be set via the ".dalirc" setup file (section 1.12).

7. HASH MODE: Set drawing mode for instances normal / hashed.

   In hashed mode the layers of the cell instances are drawn hashed, with lower intensity. Thus one can good see the difference between the layers of the edited cell and the layers of the instances of other cells in the edited cell. The hash mode can also be set via the ".dalirc" setup file (section 1.12).

8. DRAW MODE: Set drawing mode for layers transparent / dominant.

   The drawing mode can be set to either transparent or dominant. The command has only effect if a drawing order for the layers is specified. In transparent mode mix-colors are displayed if mask geometries of different layers are stacked on top of each other. In dominant mode 'top' layers hide 'lower' layers. The drawing order of the layers can be controlled via the ".dalirc" setup file (section 1.12).

9. DEF_LEVEL: Set default expansion level.

   Normally the default expansion level after the *read_cell* command (in the DB_menu) is one, however another level can be selected. If a default level of 'no' is selected, then dali is asking for a level after each *read_cell* command. The default level can also be set via the ".dalirc" setup file (section 1.12).

10. ASK_INST: Set query for instance name on / off.

    By the placing of instances of cells, the *add_inst* command (in the inst_menu), it is possible directly to ask for the instance names (if set).

11. TRACKER: Set tracker mode (X Window-version only).

    The tracker which displays coordinate information in the rightmost part of the TEXT viewport can be switched between on, off or automatic. In automatic mode, the tracker is only on when the cursor is switched to rubber-box or rubber-line cursor.

10. The VISIBLE MENU

As opposed to the other sub-menu's presented in the previous chapters, the *visible* menu actually is a dedicated menu for *one* command: the *visible* command. Upon entering this command, the LAYER viewport is used for displaying the visibility setting of the layers of the current process, and a menu is presented containing some menu items which are control commands, together with some menu items corresponding to certain classes of design data / graphical features:

disp_grid	(display grid)
snap_grid	(snap grid)
terminals	(terminals)
sub_terms	(sub-terminals)
instances	(instances)
bboxes	(bounding boxes)
term_name	(terminal names)
subt_name	(sub-terminal names)
inst_name	(instance names)

Each of the menu items can be *set* / *reset* by toggling the corresponding area. The menu items that are set are highlighted. Initially, all items are set. Pointing at an item causes it to be reset. Pointing at items that have previously been reset will set it. Upon selection of the *return* menu item, the newly chosen visibility settings will become effective and the visible menu is exited.

The other menu commands are:

— *picture*: Shows directly the effect of the current visibility setting without leaving this menu.

— *restore*: Restores the old visibility setting or the last saved one.

— *save*: Saves the current visibility setting for later use.

— *all_on*: Puts all "visibles" on.

— *all_off*: Puts all "visibles" off.

Note that the layers and the graphical features can also be set with the *visible* command via the ".dalirc" setup file.

CONTENTS

1. INTRODUCTION 1
   1.1 Running Dali 1
   1.2 The Screen Layout 2
   1.3 Pointing and Cursors 4
   1.4 Cell Editing 5
   1.5 (Non-) Orthogonal Mask Geometries 6
   1.6 Terminals 7
   1.7 The Cell Hierarchy 7
   1.8 Layers 9
   1.9 Panning and Zooming 10
   1.10 The Grid 10
   1.11 Design Rule Checking 11
   1.12 The .dalirc Setup File 12

2. THE MENU STRUCTURE 17
   2.1 The Command Classes 17
   2.2 The Main Menu 17
   2.3 Command Selection / Operation 18

3. The DB_MENU 20

4. The BOX_MENU 22

5. The TERM_MENU 27

6. The INST_MENU 30

7. The DRC_MENU 33

8. The INFO_MENU 36

9. The SETTINGS MENU 37

10. The VISIBLE MENU 39

LIST OF FIGURES

Figure 1.1. Appearance of dali with a layout being displayed 4